Learning Analytics to Assess Students' Behavior With Scratch Through Clickstream

Daniel Amo[1], Marc Alier[2], Francisco J. García-Peñalvo[3], David Fonseca[1], María J. Casañ[2]

[1] La Salle, Universitat Ramon Llull (SPAIN)
damo@salleurl.edu
[2] Universitat Politècnica de Catalunya (SPAIN)
{ludo, mjcasany}@essi.upc.edu
[3] University of Salamanca (SPAIN)
fgarcia@usal.es

Abstract. The construction of knowledge through computational practice requires to teachers a substantial amount of time and effort to evaluate programming skills, to understand and to glimpse the evolution of the students and finally to state a quantitative judgment in learning assessment. This suposes a huge problem of time and no adecuate intime feedback to students while practicing programming activities.

The field of learning analytics has been a common practice in research since last years due their great possibilities in terms of learning improvement. Such possibilities can be a strong positive contribution in the field of computational practice such as programming.

In this work we attempt to use learning analytics to ensure intime and quality feedback through the analysis of students behavior in programming practice. Hence, in order to help teachers in their assessments we propose a solution to categorize and understand students' behavior in programming activities using business technics such as web clickstream.

Clickstream is a technique that consists in the collection and analysis of data generated by users. We applied it in learning programming environments to study students behavior to enhance students learning and programming skills.

The results of the work supports this business technique as useful and adequate in programming practice. The main finding showns a first taxonomy of programming behaviors that can easily be used in a classroom. This will help teachers to understand how students behave in their practice and consequently enhance assessment and students' following-up to avoid examination failures.

Keywords: learning analytics, clickstream, scratch, programming, big data.

1 Introduction

The discipline of learning analytics allows to analyse student behavior patterns when they interact with tools and online learning environments, set them in context with their learning outcomes and draw conclusions to enhance the evaluation or improve the

learning process. Computational thinking [1] environments, such as programming ones, can be adapted to integrated a Learning Analytics system. Therefore, any of the related steps of a programming task can be analysed, namely: first actions of the student after the presentation of the activity, analysis and design of the solution, process of development and delivered code.

The results obtained by this type of analysis can be considered as very valuable information for any teacher, since it will allow them to know how each student is evolving, what their current status is in relation to the proposed tasks and what their possible risk to suspend according to trends extracted from previous analyses. Hence, the student's follow-up in a programming environment could help teachers to provide better support, enhance tutoring and adapt content or activities.

In a programming activity each student has a different style and therefore can develop a unique and different solution in comparison to the rest of their classmates. This means that the teacher must spend a great deal of time analysing each of the possible solutions delivered. In order to know what the student has done or what their behavior has been during the development of the proposed activities, we can benefit from automatic data collection and analytical techniques. The application of Learning Analytics in programming practice now makes much more sense [2-3]. Therefore, in this work we propose the use of the clickstream technique for the collection and analysis of students' interaction data.

Clickstream is a technique used in web applications, typically in e-commerce, to know how visitors behave in a website. Through the collection of clicks in the different parts of the web pages and an analytical and visualization tool you can get an idea of the behavior of the visitors. This information can be very valuable in refining the user experience and adapting business strategies to maximize the conversion of visits into sales.

We propose that teachers use this technique of data analysis in visual programming environments like Scratch. This will allow to know how students proceed in these learning and programming environments and thus improve their tutoring and support to students. The evaluation can also be improved due teachers will obtain objective information about students' deliveries.

We also propose a modification of the Scratch tool to incorporate this click collection technique. Students can develop applications in Scratch through the stacking of blocks of instructions and in a very visual. This also facilitate the learning of programming concepts. The modifications carried out allow capturing all the clicks made in this visual programming environment.

The clicks allow to reconstruct in some way everything that the student has done during the development of the solution. This allows to know interesting aspects such as if he has worked in class, which blocks have used, if he has applied the learned concepts or if he has used the structures worked in the classroom. With this information we can identify and classify the different ways of approaching the assigned programming task. The collection of clicks can discover deeper aspects of students' behavior such as different programming styles according to the clicks done in zones of high concentration of clicks.

In the following sections the work is explained in terms of the state of the art, methodology and results. The state of the art exposes how learning analytics and the study of clicks can offer a new approximation to comprehend students behavior in learning programming environments such as Scratch. The methodology section gives us the opportuinity to explain how we modified the Scratch environment and which methodologies were involved to collect students data through clicks interaction, which behavior patterns were discobered from frst big data analysis and how those patterns were positively correlated with assessments results. A conclusions section closes the paper followed by an ethics statement about students collected data.

2 Theory

Since 2010, the analysis of learning has become relevant in the field of research to improve learning and the educational context in general. There is a concern in the scientific community to understand how students learn to program from their beginnings to advanced levels [4-7]. There is no single definition although the one proposed by Erik Duval [8] that seems to be the most accurate to explain this situation when he defines "Learning Analytics is about collecting traces that learners leave behind and using those traces to improve learning".

Different proposals encourage the use of learning analytics to improve the teaching of programming. Sherman and Martin [9] propose an analytical approach to extract patterns of behavior in student developers of App Inventor projects, while other authors try to glimpse the progression of computational skills in such programming platform [10]. In comparison with our work, which also aims to identify patterns of behavior, our approach is based on clicks instead of snapshots of the source code. This is an approach not used so far in this type of programming environment that we hope can provide interesting data to the scientific-educational community.

Other authors [11] have proposed to understand how novice programmers approach their first lines of code in conventional programming environments. His approach to data collection is mixed, based on the keystrokes and events in the IDE, including clicks. However, these authors have focused on the keystrokes and have not gone deeply into the analysis of clicks.

In business it is very common to analyse processes, resources and tasks. In e-commerce, it is even an indispensable requirement. As a result, different techniques have been developed to analyse the behavior of customers. In this paper we propose the use of the clickstream technique used in business. The use of this technique is usually applied in the e-commerce web pages [12]. In this way the behavior of the clients can explain their way of acting and create a taxonomy of behaviors to offer them more related products or to guide them better through the website. In education this technique can be applied to better understand students and to personalize learning. We understand that clicks will offer additional information and complement current research.

Blikstein et al. [13] has also used educational data mining techniques and learning analytics in student code snapshots during their programming tasks. Blikstein intends to use these approaches to automate assessments that would otherwise be impossible

for teachers to detect or to be very expensive in time. Our objective and methods of analysis resemble those of this author, but the data capture differs and also the modelling of capture, prediction, analysis and visualization [14-16].

3 Methodology

In order to capture the clicks, store them, analyse them and draw conclusions, we have developed a solution supported by an architectural model based on events, a webservice to send and store the interactions and a predictive model to forecast understandable results for teachers.

3.1 Click collection

We have started the development from the original Scratch source code to create the new programming context and click capturing system.

The solution has been tested in different Scratch workshops held at La Salle Campus Barcelona during 2017/2018. Due to the limited time available from the start of the project until the first workshop, we have been able to develop a stable version, although there is still a long way to go to complete all its possibilities. However, enough interactions have been extracted to offer a first analysis and results of the tool.

Figure 1 shows the flow of technologies used for the development of the modified Scratch tool with Learning Analytics.

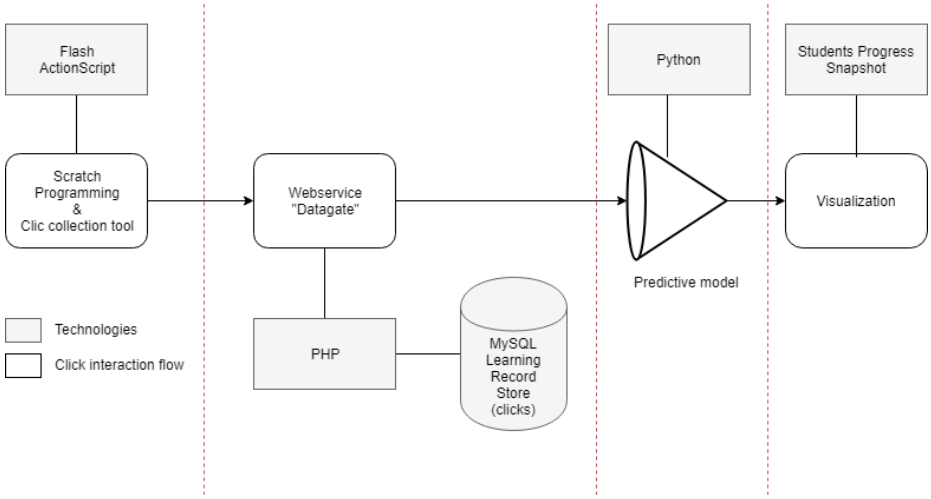

Fig. 1. Flow of technologies used for the development of the modified Scratch tool.

3.2 Analytics and predictions

Fields et al. [10] show in their research how it is possible to use learning analytics to understand how students program in Scratch. The study is based on the analysis of the

time dedicated and the developed code of the solutions to the proposed problems. Their results show different metrics in relation to application initialization and concurrent execution. We think that a new approach based on clickstream is possible to discover new behaviors.

We consider at the same time that the analysis must be accompanied by an assessment by the teaching staff to establish a relationship between qualifications and behaviors. This relationship will allow teachers to obtain information on possible risks and act accordingly. The use of rubrics will be enough to obtain a grade and analyse possible relationship with the proposed behaviors.

3.3 Behavior patterns

To comprehend where the students interacted in Scratch we created an image with the Scratch interface and the map of clicks.

Fig. 2. Zones of Scratch user interface students have clicked on.

Figure 2 shows exactly where students click on Scratch. It can be seen that students interact a lot with the "green flag" button and the "stop" button. These two buttons are used in Scratch to indicate where the code begins execution or to initialize the status of the program.

We wanted to detect the following patterns analysing the clickstream of these two buttons:

- **Blocked development (blocked).** This type of behavior defines those students who are having trouble with coding. The causes can be different, for example by distractions, by attention to non-coding aspects such as design and graphics or by a true ignorance of how to code the solution.

- **Development at a normal (normal) pace**. This type of behavior defines those students who have a balance of development between the graphic interface of the program and the coding.
- **Rapid development based on trial-error (rapid).** This type of behavior defines those students who make rapid changes in the program and constantly check the results.

The discovery of these patterns is based on the number of clicks made on the "green flag" button. Scratch allows you to execute the developed code by pressing a button identified with a green flag. This action allows to indicate in which part or parts of the code it have to start the execution, the initialization of the variables or the initialization of the program status and to provide a structured execution flow. These aspects are those that are assessed with the rubric and those that are intended to associate to the behavior patterns during development of programming activities.

We used a statistical approximation to analytically detect the three types of behavior presented –blocking, normal, rapid-. This allows us to measure the pace of development of the projects in relation to the class group. The interactions between the different groups flow spontaneously, which changes their way of developing and final results since they exchange knowledge throughout the development. Therefore, in the statistical analysis all the clicks of all the groups of each workshop are considered. Consequently, to extract the types of behavior we elaborated a statistical calculation in relation to the median of the clicks made on the green flag button.

The predictive model of behaviors is summarized in the following three points:

- Groups that click on the green flag below the 10% of the average are considered blocked developments.
- Groups that click on the green flag above the 90% of the average are considered rapid developments based on trial and error.
- The other groups are considered as development at a normal pace.

In the analysis of collected data we found a relationship between the results of the rubrics and the patterns detected. A close relationship between results and patterns will help teachers personalize student learning. For example, a low qualification in the rubric should be reflected in a blocking behavior. In some way this approach serves as an automation of the rubric through student behavior. This information can be obtained in real time. Consequently, the teacher may act before the student turns in the activity.

4 Results

Modifications in the Scratch source code have allowed to capture more than 38,000 clicks made by students among more than 30 different programming projects. The data collected includes the position of the click on the screen, the date and time of the moment of interaction, the object in which the click was made, the container of the object and the action take, such as moving, adding or deleting blocks.

The projects developed in the workshops have been evaluated by the two instruments proposed in this work. On the one hand the code of the solutions has been evaluated with a rubric. On the other hand, clicks have been introduced in the predictive model

to extract behaviors. This has allowed to obtain a score and a taxonomy for each of the projects and find a first correlation between them.

Table 1 shows the results of some of the projects assessed in terms of the rubric and behaviour.

Table 1. Projects assessed in terms of the rubric and behavior.

Project	Rubric grade	Clicks on green flag button	Behavior
Group #1	13	83	Rapid
Group #2	10	83	Rapid
Group #3	12	66	Rapid
Group #4	5	1	Blocked
Group #5	13	52	Rapid
Group #6	5	1	Blocked
Group #7	6	2	Blocked
Group #8	9	4	Blocked

Results of the analysis of the bivariate correlation of the rubric grades and the clicks on the green flag button are shown in table 3.

Table 2. Results of the analysis of the bivariate correlation considering rubric grades and clicks on the green flag.

p-value	correlation coeficient
0.02380621256650697	0.926184147506773

As observed in table 2, the correlation coefficient is close to 1. This indicates a strong correlation between the evaluation by rubric and the clicks to the green flag button. The p-value of the statistical test is less than 0.05. This indicate that there is a statistical significance. Consequently, it can be affirmed that there is a correlation between those projects in which students make rapid iterations of trial and error and good results. On the other hand, the students who make fewer executions of the program are those who have the simplest or unfinished programs.

5 Conclusions

This work has been possible thanks to the release of the Scratch source code. It has been possible to create a fork and add new software architectures that allow capturing clicks of student interactions. A first predictive analysis correlated with real assessments of the teachers shows how it is possible to analyse students behaviors in relation to the clicks made in a visual programming environment based on blocks.

The taxonomy of resulting behaviors are very valuable for teachers, who will be able to personalize learning and improve the teaching environment. Now teachers can act before students deliver incorrect or at least incomplete tasks.

We hope to add new behavior patterns with a deeper analysis of the clicks beyond the "green flag" button. Other behaviors can be differentiated through the analysis of adding, modifying or eliminating blocks in the sprites' code. This will be possible as soon as we have more data.

Our work is original in comparisson to data collection approximations of other authors studies. Berland et al. [3], Martin&Sherin[7], Sherman&Martin [9] and Vihavainen et al [11] use snapshots of students' code in different time intervals to study and extract results in their analysis. No work have fully attention to the clickstream technique. Furthermore, we tested and validated its feasibility to extract adequate behavior patterns to enhance teacher feedback. Further work will consist in compare and complement our work with the reuslts of these authors in terms of discovered pattern behaviors.

The investigation continues its course. The next phase consists in tracking students along different courses. This will allow us to identify new behaviors and provide new findings to enhance tutoring, following-up and evaluation of students.

There are limitations linked to the original Flash. We hope to be able to make an exhaustive follow-up course after course with the same students to improve the accuracy of the prediction model. This will be possible when the new version of Scratch based on HTML5 is published.

6 Ethics

The collected data does not store sensitive or personal information of students. Any possible sensitive data has been depersonalized since the same click capture. In this way, any student interaction reaches the analysis phase absolutely anonymized.

References

1. García-Peñalvo, F.J., Mendes, A.J.: Exploring the computational thinking effects in pre-university education. Comput. Human Behav. 80, 407–411 (2018).
2. Lye, S.Y., Koh, J.H.L.: Review on teaching and learning of computational thinking through programming: What is next for K-12?, (2014).
3. Berland, M. et al.: Using Learning Analytics to Understand the Learning Pathways of Novice Programmers. J. Learn. Sci. 22, 4, 564–599 (2013).
4. Grover, S.: 1e Unlocking the Potential of Learning Analytics in Computing Education.
5. Grover, S. et al.: A Framework for Using Hypothesis-Driven Approaches to Support Data-Driven Learning Analytics in Measuring Computational Thinking in Block-Based Programming Environments. ACM Trans. Comput. Educ. 17, 3, 1–25 (2017).
6. Ihantola, P. et al.: Educational Data Mining and Learning Analytics in Programming. In: Proceedings of the 2015 ITiCSE on Working Group Reports - ITICSE-WGR 15. pp. 41–63 ACM Press, New York, New York, USA (2015).

7. Martin, T., Sherin, B.: Learning Analytics and Computational Techniques for Detecting and Evaluating Patterns in Learning: An Introduction to the Special Issue. J. Learn. Sci. 22, 4, 511–520 (2013).
8. Learning Analytics and Educational Data Mining | Erik Duval's Weblog, https://erikduval.wordpress.com/2012/01/30/learning-analytics-and-educational-data-mining/.
9. Sherman, M., Martin, F.: Learning analytics for the assessment of interaction with App Inventor. In: 2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond). pp. 13–14 IEEE (2015).
10. Fields, D.A. et al.: Combining Big Data and Thick Data Analyses for Understanding Youth Learning Trajectories in a Summer Coding Camp.
11. Vihavainen, A. et al.: How novices tackle their first lines of code in an IDE. In: Proceedings of the 14th Koli Calling International Conference on Computing Education Research - Koli Calling '14. pp. 109–116 ACM Press, New York, New York, USA (2014).
12. Senecal, S. et al.: Consumers' Decision-Making Process and Their Online Shopping Behavior: A Clickstream Analysis Consumers' Decision-Making Process and Their Online Shopping Behavior: A Clickstream Analysis Consumers' Decision-Making Process and Their Online Shopping Behavior: A Clickstream Analysis. (2003).
13. Blikstein, P. et al.: Programming Pluralism: Using Learning Analytics to Detect Patterns in the Learning of Computer Programming. J. Learn. Sci. 23, 4, 561–599 (2014).
14. Gómez-Aguilar, D. A., García-Peñalvo, F. J., & Therón, R. (2014). Analítica Visual en eLearning. El Profesional de la Información, 23(3), 236-245. doi:10.3145/epi.2014.may.03
15. Gómez-Aguilar, D. A., Hernández-García, Á., García-Peñalvo, F. J., & Therón, R. (2015). Tap into visual analysis of customization of grouping of activities in eLearning. Computers in Human Behavior, 47, 60-67. doi:10.1016/j.chb.2014.11.001
16. Amo, D., Alier, M., Casany, M.J., Mayol, E.: A learning analytics tool with hybrid graphical and textual interpretation generation. Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality, pp. 327-333. ACM, Salamanca, Spain (2016)